

# On Constructing Homomorphic Encryption Schemes from Coding Theory <sup>★</sup>

Frederik Armknecht<sup>1</sup>, Daniel Augot<sup>2</sup>, Ludovic Perret<sup>3</sup>, and Ahmad-Reza Sadeghi<sup>4</sup>

<sup>1</sup> Universität Mannheim, Germany

<sup>2</sup> LIX - INRIA Saclay-Ile de France

<sup>3</sup> UPMC, University Paris 06/INRIA, France

<sup>4</sup> Technische Universität Darmstadt, Germany & Fraunhofer SIT, Germany.

**Abstract.** We introduce a generic construction principle for homomorphic encryption schemes based on coding theory. These possess several non-standard positive features. First, they are not restricted to linear homomorphism but allow for evaluating multivariate polynomials up to a fixed (but arbitrary) degree  $\mu$  on encrypted field elements. Second, they can be instantiated with various error correcting codes, even for codes with poor correcting capabilities. Third, depending on the deployed code, one can achieve very efficient schemes.

As a concrete example, we present an instantiation based on Reed-Muller codes where for  $\mu = 2$  and  $\mu = 3$  and security levels between 80 and 128 bits, all operations take less than a second (after some pre-computation). However, our analysis reveals also limitations on this approach. For structural reasons, such schemes cannot be public-key, allow for a limited number of fresh encryptions only, and cannot be combined with the bootstrapping technique. We argue why such schemes are nonetheless useful in certain application scenarios and discuss possible directions on how to overcome these issues.

**Keywords:** Homomorphic Encryption, Coding Theory, Efficiency, Provable Security.

## 1 Introduction

Homomorphic encryption schemes are very useful cryptographic tools that enable secure computation. Informally, an encryption scheme  $E_k$  is homomorphic with respect to a set of functions  $\mathcal{F}$  if for any  $f \in \mathcal{F}$  one can compute  $E_k(f(x_1, \dots, x_n))$  from  $E_k(x_1), \dots, E_k(x_n)$  without knowing  $x_1, \dots, x_n$ . Even if  $\mathcal{F}$  contains only functions based on one operation, e.g., linear operations, such schemes have been used in various applications, such as electronic voting [15, 5, 19, 20], private information retrieval (PIR) [35], oblivious polynomial evaluation (OPE) [39], or multiparty computation [18].

---

<sup>★</sup> The work described in this paper has been supported by the Commission of the European Communities through the ICT program under contract ICT-2007-216676 (ECRYPT-II). L. Perret is also supported by the french ANR under the CAC project (ANR-09-JCJC-0064-01) and the EXACTA project (ANR-09-BLAN-0371-01).

A variety of different approaches (and according hardness assumptions and proofs of security) have been investigated in the last decades. The set of considered underlying problems include the Quadratic Residuosity Problem (e.g. Goldwasser and Micali [29]), the Higher Residuosity Problem (e.g., Benaloh [5]), the Decisional Diffie-Hellman Problem (e.g., ElGamal [24], Gentry et al. [28], Prabhakarany and Rosuleky [42]), and the Decisional Composite Residuosity Class Problem (e.g. Paillier [41], Damgård and Jurik [22]). With respect to homomorphic schemes that support more than one operation, a variety of different assumptions have been considered as well: the Ideal Coset Problem (e.g. Gentry [26]), the Approximate-GCD problem (e.g. van Dijk et al. [50]), the Polynomial Coset Problem (Smart and Vercauteren [46]), the Approximate Unique Shortest Vector Problem, the Subgroup Decision Problem, the Differential Knapsack Vector Problem (all of them have been considered in Aguilar Melchor et al. [38]) To the best of our knowledge, no scheme exists so far whose the security is based on the hardness of a problem from coding theory. Such schemes would be interesting for at least the following two reasons.

*1) Alternative Security Assumption.* As the overview above indicates, most schemes are based on assumptions from number theory. For the case that these turn out to be wrong, e.g., due to quantum computers, it would be important to have schemes at hand that are based on alternative assumptions. As summed up in [3], in general promising candidates as alternatives to number theoretic problems include: the problem of solving multivariate equations over a finite field, the problem of finding a short vector in a lattice, and the problem of decoding a linear code. While homomorphic encryption schemes have been built on the first two problems. e.g., cf. [23, 38], none are known for the third.

*2) Efficient Decryption Operation.* In his seminal work [26], Gentry introduced a new approach (and an instantiation based on ideal lattices) for constructing fully-homomorphic encryption schemes, i.e., schemes that allow for evaluating any function on encrypted data. One core idea is to bootstrap a scheme. The key observation is that for getting a fully-homomorphic encryption scheme, it is actually sufficient to design a scheme such that  $\mathcal{F}$  contains the decryption operation plus some additional operation. Gentry's seminal paper [26] inspired a series of new constructions [46, 50, 9, 8] that can be bootstrapped in principle. However, all them are up to now rather proofs of concepts although some efforts have been made to improve the efficiency of the schemes [47, 27].

Smart and Vercauteren [46] estimated that to evaluate a circuit of depth  $2 - 3$  the size of the public key ranges from 262, 144 bits (with a security level of  $2^{80}$ ) to 741, 455 bits (with a security level of  $2^{100}$ ). This is actually not sufficient to achieve full-homomorphism. According to [46], this would require to evaluate deeper circuits of level 7 or 8 and to consider lattices of dimension  $2^{27}$ .

Van Dijk et al. [50] proposed a scheme that impresses with its conceptual simplicity. To the best of our knowledge, no concrete instantiation has been discussed so far. The authors give for the basic scheme an estimation of the asymptotic size of the involved parameters: for example the ciphertext size is in  $\tilde{O}(s^5)$  and the complexity of the scheme in  $\tilde{O}(s^{10})$  where  $s$  denotes the security parameter. As this variant is not bootstrappable, the authors explain a variation of the scheme that allows for bootstrapping, but at the price of a larger ciphertext.

Recently, Gentry and Halevi [27] have been able to implement all aspects of the scheme [26], including the bootstrapping functionality. They also improved [46] in the sense that the scheme only needs to consider lattices of dimension  $2^{15}$  to achieve full-homorphism. Interestingly enough, they proposed several challenges<sup>5</sup> with different level of security. It is worth to mention that the corresponding public-key size ranges from 70 MBytes for the smaller setting to 2.3 GBytes for the larger setting and the time effort for one bootstrapping operation from 30s (small setting) to 30mins (large setting).

Currently, all known schemes share the same problem: due to the inherent complexity of the corresponding decryption operation, large parameters need to be chosen to enable the bootstrapping technique, leading to impractical schemes. Therefore, one is highly interested into schemes with a decryption operation as simple as possible. This makes code-based schemes particularly interesting as for some codes decryption is simply a linear operation over the underlying field. Indeed, Gentry states by himself, that he views "a code-based construction as an interesting possibility" [25, p. 11].

**Contribution.** In this paper, we make the following contribution:

*Generic construction.* We present a generic homomorphic encryption scheme that supports the evaluation of polynomials up to a fixed but arbitrary degree  $\mu$  on encrypted field elements. The construction is based on coding theory, more precisely on evaluation codes. This comprises a large set of known codes such as Reed-Solomon codes, Reed-Muller codes, or Algebraic Geometric codes [31, 36]. Although we have to lay some additional conditions on these codes, we expect that many different instantiations are possible. Furthermore, we do not require the existence of efficient decoding algorithms (for the case that the error locations are unknown). Hence, evaluation codes that are not useful for practical error correction might be applicable for our construction.

*Impossibility Results.* The construction is based on the following (from our point of view) natural design decisions:

- 1) Ciphertexts are erroneous codewords where the secret key allows to determine the error.

<sup>5</sup> [https://researcher.ibm.com/researcher/view\\_project.php?id=1548](https://researcher.ibm.com/researcher/view_project.php?id=1548).

2) As the set of codewords forms an additive group, we preserve this property for directly achieving linear homomorphism.

We show that for structural reasons, these imply a set of limitations: (i) the scheme can never be public-key, (ii) the number of encryptions needs to be restricted, and (iii) the application of Gentry’s bootstrapping technique is not possible, thus negatively answering the raised question. In Sec. 3.4, we argue that such schemes can still be useful and discuss in Sec. 5 several strategies on how to overcome these limitations.

*Security analysis.* We show that under the assumption that the attacker *knows* the deployed code, the semantic security can be reduced to the known Decisional Synchronized Codeword Problem (DSCP). The security reduction makes use of the fact that the scheme is similar to a (generalized version) of a scheme developed by Kiayias and Yung [33]. Considering DSCP allows for determining parameter choices for concrete instantiations. Moreover, it turns out that all efficient attacks known so far on DSCP require the knowledge of the code. Thus, by keeping the code *secret* (which is possible without losing the homomorphic properties) a security margin is present in our scheme that might allow for even more efficient realizations. Indeed, the problem of recovering an unknown code from erroneous codewords only, called *Noisy Code Recognition Problem* (NCRP), has been investigated before and seems to be hard.

*Efficiency and Concrete instantiation.* As opposed to other constructions, our scheme works over finite fields. More precisely, ciphertexts are vectors of length  $n$  over some finite field and additive resp. multiplicative homomorphic operations are the component-wise sum resp. multiplication of vectors. The only condition on the underlying field is that the field size is above some lower bound. Hence, by choosing binary fields  $\text{GF}(2^\ell)$ , the basic operations (addition and multiplication of field elements) can be efficiently realized. In general, the effort for encryption is in  $\mathcal{O}(n^2)$  and the efforts for decryption, addition, and multiplication are all in  $\mathcal{O}(n)$ .

As an example, we describe an instantiation based on Reed-Muller codes and determine both concrete values and formulas for the asymptotic complexity under the assumption that the attacker knows the code, that is recovered part of the secret key already. The asymptotic ciphertext length  $n$  is in  $\mathcal{O}(\mu^3 s)$  where  $s$  is the security parameter and  $\mu$  the degree of the polynomials that can be evaluated. We present a concrete implementation where the size of the key and ciphertexts range from 591 Byte and 9.81 KByte, resp., for  $\mu = 2$  and a security level of  $2^{80}$  to 3.21 KByte and 60.95 KByte, resp., for  $\mu = 3$  and a security level of  $2^{128}$ . In all test cases, all operations take less than a second (after some pre-computation). Due to the additional security margin (gained by keeping the

code secret), more efficient parameter choices are expected to be possible (as soon as the hardness of the NCRP is better understood).

**Outline.** We provide the necessary notation and preliminaries in Sec. 2. Then, we present our generic encryption scheme in Sec. 3 and discuss some properties. In Sec. 4 we explain the underlying decoding problem and reduce the semantic security of our scheme to its hardness. In Sec. 5, we discuss a concrete instantiation based on Reed-Muller codes and analyze its efficiency. Finally, Sec. 5 concludes the paper. Due to space limitations omitted proofs and an exact description of how the parameters are derived are presented in an extended version [1].

## 2 Preliminaries

**Notation.** For an integer  $n \geq 1$ , we denote by  $[n]$  the set of integers  $1, \dots, n$ . In the following,  $s$  will be a security parameter and  $\mathbb{F}$  some arbitrary finite field. Vectors  $\mathbf{v}$  will be expressed in bold letters. We denote by  $\mathbb{F}[x_1, \dots, x_t]$  the ring of multivariate polynomials in the indeterminates  $x_1, \dots, x_t$  with coefficients over  $\mathbb{F}$ . For a polynomial  $p \in \mathbb{F}[x_1, \dots, x_t]$  and a sequence of vectors  $\mathbf{V} = (\mathbf{v}_1, \dots, \mathbf{v}_\ell) \in (\mathbb{F}^t)^\ell$ , we define  $p(\mathbf{V}) := (p(\mathbf{v}_1), \dots, p(\mathbf{v}_\ell)) \in \mathbb{F}^\ell$ .

$|\mathbf{v}|$  denotes the Hamming weight of a vector  $\mathbf{v}$ , that is the number of non-zero entries, and  $\text{supp}(\mathbf{v})$  the indices of the non-zero entries. For two vectors  $\mathbf{v}$  and  $\mathbf{w}$ , the expression  $\mathbf{v} \cdot \mathbf{w}$  stands for the component wise product: if  $\mathbf{v} = (v_1, \dots, v_n)$  and  $\mathbf{w} = (w_1, \dots, w_n)$ , then  $\mathbf{v} \cdot \mathbf{w} = (v_1 \cdot w_1, \dots, v_n \cdot w_n)$ .

A function  $f : \mathbb{N} \rightarrow \mathbb{R}$  is called negligible if for any polynomial  $p(x)$  over the real numbers there exists an integer  $n' \in \mathbb{N}$  such that  $|f(n)| < 1/p(n)$  for all  $n \geq n'$ . We sometimes write  $f = \text{negl}(n)$ . For two sets  $S$  and  $S'$  equipped of a binary operator  $\circ$ , we define  $S \circ S' := \{s \circ s' | s \in S, s' \in S'\}$ .

**Linear Codes and Evaluation Codes.** In this paper, we aim for constructing homomorphic encryption schemes based on evaluation codes. Note that such codes are a special sub-class of linear codes. First, we recall their definition:

**Definition 1 (Linear Codes).** A  $[n, k, d]$  linear code (with  $k < n$ ) is a  $k$ -dimensional linear subspace  $\mathcal{C}$  of  $\mathbb{F}^n$  with minimum Hamming distance  $d$ . That is,  $\mathbf{w} + \mathbf{w}' \in \mathcal{C}$  and  $|\mathbf{w} + \mathbf{w}'| \geq d$  for all  $\mathbf{w}, \mathbf{w}' \in \mathcal{C}$ . We call vectors  $\mathbf{w} \in \mathcal{C}$  as error-free codewords and vectors  $\mathbf{w} \in \mathbb{F}^n \setminus \mathcal{C}$  as erroneous codewords. Erroneous codewords can be written as  $\mathbf{w} + \mathbf{e}$  where  $\mathbf{e} \in \mathbb{F}^n \setminus \mathbf{0}$  is called the error vector. The bad locations, that is where an error occurred, are  $\text{supp}(\mathbf{e})$  and the good locations, i.e. error-free locations, are  $[n] \setminus \text{supp}(\mathbf{e})$ . Let  $I \subseteq [n]$ , we set:

$$\mathcal{C}(I) := \{\mathbf{w} + \mathbf{e} | \mathbf{w} \in \mathcal{C}, \mathbf{e} \in \mathbb{F}^n, \text{supp}(\mathbf{e}) \subseteq [n] \setminus I\}. \quad (1)$$

In other words,  $\mathcal{C}(I)$  is the set of all erroneous codewords where the positions specified in  $I$  are good locations. In particular, we have  $\mathcal{C}([n]) = \mathcal{C}$ .

In addition, there exist two efficient (that is polynomial in the input length) algorithms  $\text{Encode} : \mathbb{F} \rightarrow \mathcal{C}$  (encoding) and  $\text{Decode} : \mathbb{F}^n \times \{I \mid I \subseteq [n]\} \rightarrow \mathbb{F}$  (decoding) such that  $\text{Decode}(\text{Encode}(m) + \mathbf{e}, \text{supp}(\mathbf{e})) = m$  for all  $m \in \mathbb{F}$  and  $\mathbf{e} \in \mathbb{F}^n$  with  $|\mathbf{e}| < d$ . This means that erroneous codewords can be decoded efficiently possible if the bad locations are known.<sup>6</sup>

By choosing an encryption scheme where the ciphertexts are codewords from a linear code and where the errors are at fixed, but secret locations  $I$ , yields directly a linear homomorphic encryption scheme. However, to get multiplicative homomorphism as well, we need codes such that the product of two codewords is a codeword again. For this reason, we focus on a special class of linear codes, called evaluation codes.

**Definition 2 (Evaluation Codes).** Let  $\mathcal{X}$  be a geometric object<sup>7</sup> together with a tuple  $\mathbf{x} := (x_1, \dots, x_n)$  of  $n$  distinct points in  $\mathcal{X}$ . Let  $\mathbb{F}^{\mathcal{X}}$  denote the set of all mappings from  $\mathcal{X}$  to  $\mathbb{F}$ . We assume that we have a vector space  $(\mathcal{L}, +)$  over  $\mathbb{F}$ , with  $\mathcal{L} \subseteq \mathbb{F}^{\mathcal{X}}$ , of dimension  $k$ . An evaluation code  $\mathcal{C}$  is obtained by using  $\text{Encode} = \text{ex} \circ \text{ev}$  where

**Expression:**  $\text{ex} : \mathbb{F} \rightarrow \mathcal{L}, m \mapsto p$  is a mapping that expresses a message  $m \in \mathbb{F}^{k'}$  (with  $k' \leq k$ ) as a function  $p \in \mathcal{L}$  and

**Evaluation:**  $\text{ev}_{\mathbf{x}} : \mathcal{L} \rightarrow \mathbb{F}^n, p \mapsto p(\mathbf{x}) = (p(x_1), \dots, p(x_n))$  is the evaluation mapping that maps a function  $p \in \mathcal{L}$  to its evaluation on  $\mathbf{x}$ .

We call  $\mathcal{L}(\mathcal{C}) := \mathcal{L}$  the function space of  $\mathcal{C}$  and the vector  $\mathbf{x}$  the codeword support. For a codeword  $\mathbf{w} = p(\mathbf{x})$ ,  $p_{\mathbf{w}} := p$  is the solution function of  $\mathbf{w}$ .

Evaluation codes are useful for our goal as (under certain conditions) the product of two codewords is a codeword again. Let the product of two functions  $p, p' \in \mathbb{F}^{\mathcal{X}}$  be defined by  $(p \cdot p')(x) := p(x) \cdot p'(x) \in \mathbb{F}$  for all  $x \in \mathcal{X}$ . Now assume two codes  $\mathcal{C}$  and  $\hat{\mathcal{C}}$  such that  $p_1 \cdot p_2 \in \mathcal{L}(\hat{\mathcal{C}})$  for all  $p_1, p_2 \in \mathcal{L}(\mathcal{C})$ . Then, it holds for any two codewords  $\mathbf{w}_1, \mathbf{w}_2 \in \mathcal{C}$ :

$$\underbrace{\mathbf{w}_1}_{\in \mathcal{C}} \cdot \underbrace{\mathbf{w}_2}_{\in \mathcal{C}} = \underbrace{p_{\mathbf{w}_1}(\mathbf{x})}_{\in \mathcal{L}(\mathcal{C})} \cdot \underbrace{p_{\mathbf{w}_2}(\mathbf{x})}_{\in \mathcal{L}(\mathcal{C})} = \underbrace{(p_{\mathbf{w}_1} \cdot p_{\mathbf{w}_2})(\mathbf{x})}_{\in \mathcal{L}(\hat{\mathcal{C}})} \in \hat{\mathcal{C}}. \quad (2)$$

We generalize this concept in the following definition:

<sup>6</sup> This is a natural prerequisite for any practical error correction codes. Moreover, as we do not assume efficient decoding if the bad locations are *unknown*, some codes may be suitable for our scheme although they are not interesting for error correction.

<sup>7</sup> Typically,  $\mathcal{X}$  will be the affine line  $\mathbb{F}$  (leading to Reed-Solomon codes), or a Cartesian product of  $\mathbb{F}$  (leading to Reed-Muller codes).

**Definition 3 ( $\mu$ -multiplicative Codes).** An evaluation code  $\mathcal{C}$  is  $\mu$ -multiplicative if there exists a code  $\widehat{\mathcal{C}}$  with the same codeword support such that  $\mathcal{L}(\mathcal{C})^\ell \subseteq \mathcal{L}(\widehat{\mathcal{C}})$  for all  $\ell \in [\mu]$ . We use the notation  $\mathcal{C}^\mu \subseteq \widehat{\mathcal{C}}$  in this case.

*Remark 1.* The multiplicativity<sup>8</sup> of codes as such has been studied extensively in a series of papers, e.g. [12, 13, 43, 21, 44] but motivated by a different application, i.e. secure multi-party computation. In particular, [21, 44] proposed a new general framework to construct codes such that their powers are “good” in some sense (asymptotic properties, reconstruction parameters, ...). At first glance, such framework might allow to further enrich the types of codes that could be used as a platform for our scheme. This question will be addressed in an extended version of the paper.

We will later show that such codes exist in practice. Using  $\mu$ -multiplicative codes, we get a situation where the sum and the product of codewords yield a codeword again. However, this does not automatically imply that decoding the product of two codewords yields the product of the underlying messages. This actually depends on how the expression mapping “ex” in Def. 2 is realized. One approach is to incorporate the evaluation of functions here as well. More precisely, we will use in the paper the following instantiation of evaluation codes that we will call *special evaluation codes*:

**Definition 4 (Special Evaluation Codes).** A special evaluation code is an evaluation code as defined in Def. 2 where the expression mapping “ex” is realized as follows. We fix an element  $\mathbf{y} \in \mathcal{X}$ , called the message support, whose role is to encode a message into a codeword. We require that  $\mathbf{y}$  is distinct from the entries of the codeword support  $\mathbf{x}$ . For any given message  $m \in \mathbb{F}$ , we define a subset of  $\mathcal{L}$  as follows:  $\mathcal{L}_{\mathbf{y} \mapsto m} := \{p \in \mathcal{L} \mid p(\mathbf{y}) = m\}$ . We assume that there exists for any  $m$  a function  $p \in \mathcal{L}$  such that  $p(\mathbf{y}) = m$ . That is,  $\mathcal{L}_{\mathbf{y} \mapsto m}$  is non-empty. Given this, we consider an expression function ex that on input  $m$  outputs a random  $p \in \mathcal{L}_{\mathbf{y} \mapsto m}$ .

From now on, we will consider special evaluation codes only. Observe that we consider here only encoding of one field element although up to  $k$  elements would be possible. The main reason is that it simplifies some security arguments and ensures that there exist many different encodings for the same  $m \in \mathbb{F}$  which provides probabilistic encoding (= encryption). This is a necessary prerequisite for secure algebraic homomorphic encryption (see Boneh and Lipton [7]). We leave the extension to larger plaintext spaces to future work.

---

<sup>8</sup> We thank R. Cramer for pointing us this connection.

The next theorem shows that  $\mu$ -multiplicative codes allow for additive and multiplicative homomorphism regarding the encoding operation. The proof can be found in [1].

**Theorem 1 (Additive and Multiplicative Homomorphism).** *Let  $\mathcal{C}^\mu \subseteq \widehat{\mathcal{C}}$  for some special evaluation codes  $\mathcal{C}$  and  $\widehat{\mathcal{C}}$ . We consider a selection of at most  $\mu$  encodings  $\mathbf{w}_j \in \mathcal{C}(I_j)$  of messages  $m_j \in \mathbb{F}$  where  $I_j \subseteq [n]$  and set  $I := \bigcap_j I_j$ . It holds*

**Closed under addition and multiplication:**  $\sum_{j=1}^\ell \mathbf{w}_j \in \widehat{\mathcal{C}}(I)$ , and  $\prod_{j=1}^\ell \mathbf{w}_j \in \widehat{\mathcal{C}}(I)$ .

**Additive homomorphism:**  $\text{Decode}(\sum_{j=1}^\ell \mathbf{w}_j, I) = \sum_{j=1}^\ell m_j$ , if  $|I|$  good locations are sufficient for unique decoding.

**Multiplicative homomorphism:**  $\text{Decode}(\prod_{j=1}^\ell \mathbf{w}_j, I) = \prod_{j=1}^\ell m_j$ , if  $|I|$  good locations are sufficient for unique decoding.

### 3 The Encryption Scheme

#### 3.1 Description

In this section, we formally describe the construction of our encryption scheme. The scheme is symmetric and encrypts plaintexts  $m \in \mathbb{F}$  to erroneous codewords  $\mathbf{c} \in \mathcal{C}(I)$  where the key consists of  $I$ , the set of good locations, and the used supports (which in fact determines the used code instantiations). Regarding its homomorphic properties, it permits unlimited number of additions and a fixed but arbitrary number of multiplications. The scheme is composed of five algorithms: Setup, Encrypt, Decrypt, Add, Mult.

–  $(\mathcal{C}, \widehat{\mathcal{C}}, I) \leftarrow \text{Setup}(s, \mu, L)$  : The input are three positive integers  $s, L$ , and  $\mu$  where  $s$  denotes the security parameter,  $L$  the expected total number of encryptions<sup>9</sup>, and  $\mu$  the maximum degree of the supported polynomials. The Setup algorithm chooses a codeword support  $\mathbf{x}$ , a message support  $\mathbf{y}$ , and two special evaluation codes  $\mathcal{C}$  and  $\widehat{\mathcal{C}}$  such that  $\mathcal{C}^\mu \subseteq \widehat{\mathcal{C}}$  and the length of the codewords is at least  $L$ . How to choose appropriate codes and parameters highly depends on the considered coding scheme. We will describe in Sec. 5 a concrete instantiation where this is possible. Setup also generates a set  $I \subset [n]$  of size  $T$  where  $T$  depends on the parameter from above and the deployed code.  $I$  denotes the good locations for the generated encryptions and represents the secret key of the scheme. The output is the secret key  $\mathbf{k} = (\mathbf{x}, \mathbf{y}, I)$ .

–  $(\mathbf{c}, 1) \leftarrow \text{Encrypt}(\mathbf{m}, \mathbf{k})$  : The inputs are a plaintext message  $m \in \mathbb{F}$  and a secret key  $\mathbf{k} = (\mathbf{x}, \mathbf{y}, I)$ . Encrypt first chooses a random encoding  $\mathbf{w} \in \mathcal{C}$  of  $m$ , using the Encode algorithm and the knowledge of the supports  $\mathbf{x}$  and  $\mathbf{y}$ . Then,

<sup>9</sup> This means an upper bound on the value on how many messages are going to be “freshly” encrypted. It does not include the number of possible combinations of existing ciphertexts.



it samples a uniformly random error vector  $\mathbf{e} \in \mathbb{F}^n$  such that  $\text{supp}(\mathbf{e}) \subseteq [n] \setminus I$  and computes  $\mathbf{c} := \mathbf{w} + \mathbf{e}$ . Finally, the ciphertext is defined as the pair  $(\mathbf{c}, 1)$  where the first entry is an erroneous codeword in  $\mathcal{C}(I)$  that encodes the plaintext  $m$  while the second entry, the integer, is a counter to keep track of the number of multiplications.

–  $\mathbf{m} \leftarrow \text{Decrypt}((\mathbf{c}, \gamma), \mathbf{k})$  : Decrypt gets as input the secret key  $\mathbf{k} = (\mathbf{x}, \mathbf{y}, I)$  and a pair  $(\mathbf{c}, \gamma)$  with  $\mathbf{c} \in \mathcal{C}(I)$  and  $\gamma \leq \mu$ . It returns  $m := \text{Decode}(\mathbf{c}, I)$  where Decode is used with respect to  $\mathbf{x}$  and  $\mathbf{y}$ .

–  $(\mathbf{c}'', \gamma'') \leftarrow \text{Add}((\mathbf{c}, \gamma), (\mathbf{c}', \gamma'))$  : Add gets as input two ciphertexts  $(\mathbf{c}, \gamma)$  and  $(\mathbf{c}', \gamma')$  and generates an encryption of the sum of the plaintexts  $(\mathbf{c} + \mathbf{c}', \max(\gamma, \gamma'))$ .

–  $(\mathbf{c}'', \gamma'') \leftarrow \text{Mult}((\mathbf{c}, \gamma), (\mathbf{c}', \gamma'))$  : This procedure gets as input two ciphertexts  $(\mathbf{c}, \gamma)$  and  $(\mathbf{c}', \gamma')$  with  $\gamma + \gamma' \leq \mu$  and generates an encryption of the product of the plaintexts  $(\mathbf{c} \cdot \mathbf{c}', \gamma + \gamma')$  where “ $\cdot$ ” is the componentwise vector product as explained in Section 2.

**Correctness.** The correctness for the case that neither Add nor Mult have been used can be checked straightforwardly. Let  $(\mathbf{c}, \gamma)$  and  $(\mathbf{c}', \gamma')$  be two encryptions of  $m$  and  $m'$ , respectively, with  $\mathbf{c}, \mathbf{c}' \in \mathcal{C}(I)$ . Then, by Theorem 1 it follows that  $\mathbf{c} + \mathbf{c}' \in \mathcal{C}(I)$  and that  $\text{Decode}(\mathbf{c} + \mathbf{c}', I) = m + m'$ . Analogously, we have  $\mathbf{c} \cdot \mathbf{c}' \in \mathcal{C}(I)$  and  $\text{Decode}(\mathbf{c} \cdot \mathbf{c}', I) = m \cdot m'$  by Theorem 1 if  $\mathbf{c} \cdot \mathbf{c}'$  does not result from more than  $\mu$  multiplications. Observe that it is not necessary to know the deployed codes, more precisely the supports  $\mathbf{x}$  and  $\mathbf{y}$ , for executing Add or Mult. Only the knowledge of the underlying field is mandatory.

### 3.2 Alternative Description in Terms of Linear Algebra

Observe that the scheme actually represents a group homomorphic encryption scheme between the additive groups  $\mathbb{F}$  and  $\mathbb{F}^n$ . This allows an alternative, possibly simpler description in terms of linear algebra (cf. Armknecht et al. [2] for a comprehensive treatment). Let  $V_0 \subset \mathbb{F}^n$  denote the subspace that contains all encryptions of 0, let  $\mathbf{c}^* \in \mathbb{F}^n$  denote an arbitrary encryption of  $1 \in \mathbb{F}$ , and let  $V_{\text{err}} \subset \mathbb{F}^n$  denote the vector space of the error vectors. Then, encryption can be equivalently expressed by: given a plaintext  $m \in \mathbb{F}$ , sample  $v_0 \in V_0$  and  $v_e \in V_{\text{err}}$  and output  $\mathbf{c} := v_0 + m \cdot \mathbf{c}^* + v_e$ . Furthermore, there exists a vector  $v_{\text{key}} \in \mathbb{F}^n$  such that decryption can be performed as follows: given a ciphertext  $\mathbf{c}$ , compute  $m = \mathbf{c}^t \times v_{\text{key}}$  where  $\times$  denotes the usual matrix-vector product.

### 3.3 Effort and Limitations.

**Effort.** One benefit of the proposed scheme is that encryption and decryption processes are simple operations. Assume in the following that bases for  $V_0$  and  $V_{\text{err}}$  are known and likewise the vector  $v_{\text{key}}$ . In some cases, these can be directly written down. Alternatively, they can be computed by solving systems of linear

equation based on outputs from the scheme, giving an effort in  $\mathcal{O}(n^3)$  in the worst case. Using this and the description from Sec. 3.2, one can easily validate that the efforts for encryption and decryption are in  $\mathcal{O}(n^2)$  and  $\mathcal{O}(n)$ , respectively. Furthermore, Add and Mult are simply the componentwise addition and multiplication, respectively, of two vectors of length  $n$ . Thus, both operations are in  $\mathcal{O}(n)$  as well. Furthermore, the schemes work over any (finite) field as long as the field size is not too small. In particular, this allows to use computer-friendly fields  $\text{GF}(2^\ell)$  for a further speed-up.

**Structural Limitations.** The fact that the ciphertexts are in  $\mathbb{F}^n$  yields several limitations as we will elaborate now. For example an attacker who has  $n$  pairs of plaintext/ciphertext at her disposal can compute the vector  $v_{\text{key}}$  from Sec. 3.2 on her own by solving a system of linear equations. This imposes a natural upper bound on the number of encryptions: not more than  $n - 1$  known plaintexts should be encrypted under the same key as otherwise security is lost.<sup>10</sup> This also shows that Gentry’s bootstrapping technique cannot be applied. As the key  $v_{\text{key}}$  is composed of  $n$  field elements and as each ciphertext represents the encryption of one field element, one would require at least  $n$  ciphertexts, thus exceeding the bound explained above.

For similar reasons, no public-key variant can exist. Assume the opposite and let  $V'_0$  denote the set of all possible encryptions of  $0 \in \mathbb{F}$  (including linear combinations). One checks easily that  $V'_0$  forms a sub-vectorspace of  $\mathbb{F}^n$ . Furthermore, one can show (e.g., cf. [2]) that the scheme is IND-CPA secure if and only if it is hard to decide for a given vector  $\mathbf{v} \in \mathbb{F}^n$  whether  $\mathbf{v} \in V'_0$  (also called the *subspace membership problem* (SMP)). However, the latter is easy in this case. Given sufficiently many samples from  $V'_0$ , one can compute a basis and then solve the SMP using basic linear algebra techniques.

Observe that all limitations are consequences of the fact that the underlying algebraic structure is  $\mathbb{F}^n$ . Thus, by switching over to another structure, e.g., a ring, it might be possible to neutralize these.

### 3.4 Possible Applications

Before we analyze the security of the schemes, we argue why such schemes are interesting for certain applications despite their limitations. Let us consider some examples. In the Private Information Retrieval scenario a user wants to retrieve data from an untrusted (honest but curious) server without the server being able to learn the data. If the stored data would be encrypted with an appropriate homomorphic encryption scheme, applying polynomials on the encrypted

<sup>10</sup> Here, we have to emphasize that this bound holds for “fresh” encryptions only. As the combination of existing ciphertexts does not provide any new information, there are no limits there (except on the number of multiplications of course).

data could be used to filter out relevant data without revealing any useful information about the plaintext to the server. Another example applies to Oblivious Polynomial Evaluation where one party  $A$  holds a secret input  $x$  and another party  $B$  a secret polynomial  $P$ . The goal is to allow  $A$  to obtain  $P(x)$  such that no party learns anything about the secret of the other party (beyond what can already be deduced from  $x$  and  $P(x)$ ). Although solutions based on additively homomorphic encryption schemes has been discussed for the case of univariate polynomials, these become highly inefficient for the multivariate case. The deployment of schemes that support both operations would help to enormously reduce the effort.

Observe that both examples mentioned above have in common that it would be sufficient if the deployed scheme is symmetric and supports only a limited number of multiplications and a limited number of encryptions. Of course a variety of other two party applications can be envisaged where deploying such schemes is sufficient as long as there is a gain in efficiency. In other words, the crucial factor here is not *flexibility* but *efficiency*.

## 4 Security Reduction to Coding Theory

In this section, we discuss the security of the proposed scheme. Recall that according to the limitations due to the usage of  $\mathbb{F}^n$ , it is necessary to use the scheme as a symmetric-key encryption scheme and that the number of encryptions is limited to some value below  $n$ . In this section, we show that under these conditions, a security reduction to a known decoding problem is possible.

**Decisional Synchronized Codewords Problem (DSCP).** Recall that a ciphertext is actually a codeword where the location of the good positions *and* the deployed code instantiation (more precisely the used supports) form the secret key. Especially the second property is rather unusual, making it difficult to estimate the security without long-term research. To assess this question to some extent, we follow the usual approach by investigating a weaker variant of the scheme.<sup>11</sup> More precisely, we consider a variant where the used code is known to the attacker, i.e., the attacker is given (eventually) a basis for the vector space  $V_0$  while  $V_{\text{err}}$  remains secret. This allows for the following insights:

1. The security of the weaker scheme can be reduced to a known decoding problem, called Decisional Synchronized Codewords Problem (DSCP).
2. We give an overview of the most efficient algorithms for solving DSCP so far. It will turn out that all of them need to know the used code instantiation.

---

<sup>11</sup> This is in line with other approaches where, for example, attacks on reduced-round versions of a block cipher are presented and then to argue that adding more rounds give a hopefully sufficient security margin.

Furthermore, as we will argue later, the problem of figuring out the code seems to be hard according to the current state of knowledge. This indicates that keeping the code secret provides an additional security margin. Whether this margin is sufficient is subject to future research.

Similar to the the Kiayias-Yung-scheme [33], recovering the plaintext from *one* ciphertext of the weaker scheme without knowing the secret key is equivalent to decoding an erroneous codeword. But in contrast to [33], where the error locations alter from encryption to encryption, in our scheme the positions of the error free entries remain the same for all encryptions. This is a necessary prerequisite for the homomorphic property. Therefore, recovering the plaintexts from *several* ciphertexts (under the assumption that the code is known) is equivalent to decoding *several* codewords where the errors are always *at the same locations*. This is a special decoding problem which is known as the decoding synchronized codes problem (DSCP). For a formal definition, note that the problem of decoding codes is equivalent to identifying the good (resp. bad) locations: once a codeword is decoded, the good and bad locations can be easily identified, and vice versa. We use this fact for defining the following problem (see also [33]).

**Definition 5. [Decisional Synchronized Codewords Problem (DSCP)]** Let  $\tilde{\mathcal{C}}$  denote a  $[n, k, d]$  code. We consider a sampler  $\mathcal{S}$  that on input  $(\tilde{\mathcal{C}}; T, L)$  proceeds as follows:

1. Choose uniformly random a subset  $I \subset [n]$  of size  $T$
2. Sample  $L$  pairwise distinct codewords  $\tilde{\mathbf{w}}_1, \dots, \tilde{\mathbf{w}}_L \in \tilde{\mathcal{C}}$ .
3. Sample  $L$  pairwise distinct vectors  $\mathbf{e}_\ell \in \mathbb{F}^n$  such that  $\text{supp}(\mathbf{e}_\ell) = [n] \setminus I$ .
4. Compute  $\mathbf{c}_\ell := \tilde{\mathbf{w}}_\ell + \mathbf{e}_\ell \in \mathbb{F}^n$ .
5. Output  $\tilde{\mathbf{C}} := (\mathbf{c}_1, \dots, \mathbf{c}_L) \in (\mathbb{F}^n)^L$ .

In addition, we consider two modifications of  $\mathcal{S}$ ,  $\mathcal{S}^{\text{bad}}$  and  $\mathcal{S}^{\text{good}}$ .  $\mathcal{S}^{\text{bad}}$  operates analogously to  $\mathcal{S}$  but chooses in addition  $i$  at random from the set  $[n] \setminus I$ , and outputs  $(i; \tilde{\mathbf{C}})$ .  $\mathcal{S}^{\text{good}}$  is defined similarly but  $i$  is selected at random from the set  $I$  instead. We call an output  $(i; \tilde{\mathbf{C}})$  as DSCP instance.

The Decisional Synchronized Codewords Problem DSCP is to distinguish between the two sampler  $\mathcal{S}^{\text{bad}}$  and  $\mathcal{S}^{\text{good}}$ . For any probabilistic polynomial-time (PPT) algorithm  $\mathcal{A}$  we define  $\text{Adv}_{\tilde{\mathcal{C}}; T, L}^{\text{DSCP}, \mathcal{A}} :=$

$$\left| \Pr[\mathcal{A}(\mathcal{S}^{\text{good}}(\tilde{\mathcal{C}}; T, L)) = 1] - \Pr[\mathcal{A}(\mathcal{S}^{\text{bad}}(\tilde{\mathcal{C}}; T, L)) = 1] \right| \quad (3)$$

where the probability is taken over the random coins from  $\mathcal{A}$  and the samplers. The  $\text{DSCP}[\tilde{\mathcal{C}}; T, L]$  assumption holds with respect to a security parameter  $s$  if  $\text{Adv}_{\tilde{\mathcal{C}}; T, L}^{\text{DSCP}}(s) := \max_{\mathcal{A}} \text{Adv}_{\tilde{\mathcal{C}}; T, L}^{\text{DSCP}, \mathcal{A}}$  is negligible in  $s$ .

**Security Reduction.** Next, we prove that the semantic security of the weaker scheme can be reduced to the hardness of DSCP with appropriate parameters. In a nutshell, semantic security requires that it should be infeasible for an adversary to gain, for a given ciphertext, any partial information about the underlying plaintext, even if the set of possible plaintexts is reduced to two different messages which have been chosen by the attacker before. This is formally captured by a game where the adversary can request encryptions of adaptively chosen plaintexts. Her task is to tell apart encryptions of two plaintexts that were chosen by her. For a formal definition, we refer to [33].

For the proof of security, we make use of the following theorem on the pseudorandomness of sampled instances:

**Theorem 2.** *For any distinguisher  $\mathcal{A}$  between the distributions  $\mathbb{D}_{\check{\mathcal{C}};T,L}$  (induced by the sampler  $\mathcal{S}$  from Definition 5) and the uniform distribution  $\mathbb{U}_n^L$  on  $(\mathbb{F}^n)^L$ , it holds that  $|Pr[\mathcal{A}(\check{\mathcal{C}}) = 1 | \check{\mathcal{C}} \leftarrow \mathbb{D}_{\check{\mathcal{C}};T,L}] - Pr[\mathcal{A}(\check{\mathcal{C}}) = 1 | \check{\mathcal{C}} \leftarrow \mathbb{U}_n^L]|$*

$$\leq \frac{T \cdot L \cdot (n - T + 3)}{|\mathbb{F}|} + T \cdot Adv_{\check{\mathcal{C}}^-,T,L}^{\text{DSCP}} + 8T \cdot Adv_{\check{\mathcal{C}};T,L}^{\text{DSCP}}. \quad (4)$$

where  $\check{\mathcal{C}}^-$  denotes the code obtained from  $\check{\mathcal{C}}$  by removing one point from the support  $\mathbf{x}$ .

This theorem is an adaptation of Theorem 3.4 given in [33]. Despite of some subtle differences due to the fact that we are dealing with a DSCP instance here, the proof is very similar. For this reason and because of space limitation, we omit the proof and refer to the full version of this paper and/or [33].

For the reduction, we need that an attacker is able to randomize codewords in a controlled way. This can be done if the considered code allows for a special encoding of  $0 \in \mathbb{F}$  as specified now:

**Definition 6 (Special Encoding of 0).** *A special evaluation code  $\check{\mathcal{C}}$  allows for a special encoding of 0 if there exists a codeword  $\check{\mathbf{w}} = (\check{w}_1, \dots, \check{w}_n) \in \check{\mathcal{C}}$  such that  $\check{w}_i \neq 0$  for all  $i \in [n]$  and  $\text{Decode}(\check{\mathbf{w}}) = 0$ .*

We will show later a concrete instantiation, namely Reed-Muller codes, that allows a special encoding of 0. Furthermore, it is easy to show that a special encoding of 0 exists for Reed-Solomon codes. Given a special encoding of 0, one can construct a transformation on vectors that will play a crucial role in the reduction.

**Proposition 1 (Transformation).** *Let  $\check{\mathcal{C}}^2 \subseteq \mathcal{C}$  be such that  $\check{\mathcal{C}}$  permits a special encoding of 0, i.e., there exists a codeword  $\check{\mathbf{w}} = (\check{w}_1, \dots, \check{w}_n) \in \check{\mathcal{C}}$  such that  $\check{w}_i \neq 0$  for all  $i \in [n]$  and  $\text{Decode}(\check{\mathbf{w}}) = \mathbf{0}$ . Then, there exists a probabilistic mapping  $\tau : \mathbb{F}^n \times \mathbb{F}^k \rightarrow \mathbb{F}^n$  such that it holds for all  $m \in \mathbb{F}^k$ :*

1. For a uniformly random vector  $\mathbf{v} \in \mathbb{F}^n$ , the output  $\tau(\mathbf{v}, m)$  is uniformly random in  $\mathbb{F}^n$  as well.
2. For a codeword  $\check{\mathbf{w}} \in \check{\mathcal{C}}(I)$ , the output  $\mathbf{w} := \tau(\check{\mathbf{w}}, m)$  is a codeword in  $\mathcal{C}(I)$ , that is the error-free locations of  $\check{\mathbf{w}}$  and  $\mathbf{w}$  are the same. It holds that  $p_{\mathbf{w}}(\mathbf{y}) = m$ , that is  $\mathbf{w}$  decodes to the second input  $m$ , and  $\mathbf{w}$  is uniformly distributed in the set of encodings of  $m$ .

In other words,  $\tau(\cdot, m)$  transforms (erroneous) encodings under  $\check{\mathcal{C}}$  of arbitrary (possibly unknown) messages to (erroneous) encodings of  $m$  under  $\mathcal{C}$  and transforms otherwise random vectors to random vectors.

The proof of this result is given in [1].

**Theorem 3.** *The encryption scheme from Section 3 is semantically secure for parameters  $(\mathcal{C}, \hat{\mathcal{C}}; T, L)$  if the  $\text{DSCP}[\check{\mathcal{C}}; T, L]$  and the  $\text{DSCP}[\check{\mathcal{C}}^-; T, L]$  assumptions hold for some code  $\check{\mathcal{C}}$  with  $\check{\mathcal{C}}^2 \subseteq \mathcal{C}$  that has a special encoding of 0 and if  $\frac{T \cdot L \cdot (n - T + 3)}{|\mathbb{F}|} = \text{negl}(s)$ .*

*Proof.* Let  $\mathcal{A}^{SS}$  be a PPT algorithm that breaks the semantic security for parameters  $(\mathcal{C}, \hat{\mathcal{C}}; T, L)$  with at most  $L$  queries (including the challenge). We show how to turn  $\mathcal{A}^{SS}$  directly into a distinguisher  $\mathcal{A}^{dst}$  which distinguishes between the two distributions specified in Th. 2. If both assumptions  $\text{DSCP}[\check{\mathcal{C}}; T, L]$  and  $\text{DSCP}[\check{\mathcal{C}}^-; T, L]$  hold and if  $\frac{T \cdot L \cdot (n - T + 3)}{|\mathbb{F}|} = \text{negl}(s)$ , then it follows from equation (4) in Theorem 2 that the advantage of  $\mathcal{A}^{dst}$  is negligible. Consequently, this must be true for  $\mathcal{A}^{SS}$  as well which proves the semantic security.

Let  $\check{\mathbf{C}} = (\check{\mathbf{w}}_1, \dots, \check{\mathbf{w}}_L) \in (\mathbb{F}^n)^L$  be given to  $\mathcal{A}^{dst}$  which is either distributed according to  $\mathbb{D}_{\check{\mathcal{C}}, T, L}$  or according to  $\mathbb{U}_n^L$ .  $\mathcal{A}^{dst}$  now simulates the encryption oracle  $\mathcal{O}^{SS}$  for  $\mathcal{A}^{SS}$  as follows. As  $\check{\mathcal{C}}$  has a special encoding of 0, we can use the transformation  $\tau$  defined in Prop. 1. For each encryption query  $m_\ell$  from  $\mathcal{A}^{SS}$ ,  $\mathcal{A}^{dst}$  responds with  $\mathbf{c}_\ell := \tau(\check{\mathbf{w}}_\ell, m_\ell)$ . When  $\mathcal{A}^{SS}$  makes its challenge  $(\tilde{m}_0, \tilde{m}_1) \in \mathbb{F}^2$ ,  $\mathcal{A}^{dst}$  uniformly random picks  $b \in \{0, 1\}$  and responds with  $\tilde{\mathbf{c}} := \tau(\check{\mathbf{w}}_\ell, \tilde{m}_b)$  for some unused codeword  $\check{\mathbf{w}}_\ell$ .

If  $\check{\mathbf{C}}$  is distributed according to  $\mathbb{U}_n^L$ , then each  $\check{\mathbf{w}}_\ell$  is uniformly random from  $\mathbb{F}^n$  and hence each response  $\mathbf{c}_\ell$  is some random vector in  $\mathbb{F}^n$  as well by Proposition 1. In particular, the response  $\tilde{\mathbf{c}}$  from  $\mathcal{A}^{dst}$  to the challenge of  $\mathcal{A}^{SS}$  is independent of the challenge  $(\tilde{m}_0, \tilde{m}_1)$ . Thus  $\mathcal{A}^{SS}$  gains no information on the value of  $b$  which shows that its advantage is zero in this case.

Now assume that  $\check{\mathbf{C}}$  is distributed according to  $\mathbb{D}_{\check{\mathcal{C}}, T, L}$ . That is  $\check{\mathbf{w}}_\ell \in \check{\mathcal{C}}(I)$  for a common set  $I$  of good locations. By Proposition 1, each response  $\mathbf{c}_\ell$  is an encoding of  $m_\ell$  in  $\mathcal{C}(I)$ . Furthermore, this procedure yields a uniformly random encryption of a given plaintext (by Prop. 1). Therefore,  $\mathcal{A}^{SS}$ 's view is that it

received valid encryptions and any encryption for a chosen plaintext is possible. Hence, it observes no difference to communicating with an encryption oracle  $\mathcal{O}^{SS}$ . In particular,  $\mathcal{A}^{SS}$  has by assumption a non-negligible advantage to guess  $b$  correctly.

The remainder of the proof follows the usual arguments.  $\mathcal{A}^{dst}$  runs  $\mathcal{A}^{SS}$  sufficiently often to estimate  $\mathcal{A}^{SS}$ 's advantage with sufficient precision. If the advantage is negligible,  $\mathcal{A}^{dst}$  assumes that  $\mathbf{C}$  was uniformly sampled from  $(\mathbb{F}^n)^L$ . Otherwise, it assumes that  $\mathbf{C}$  was sampled by  $\mathbb{D}_{\tilde{\mathcal{C}};T,L}$ .  $\square$

**On the Gap between the Known Code and Secret Code Cases.** Obviously, additionally knowing the code ease the attacks explained in Sec. 3.3. For example, as the attacker can use her additional knowledge to compute a basis of  $V_0$  and a choice for  $\mathbf{c}^*$ , getting about  $\dim(V_{\text{err}})$  ciphertexts would be sufficient for deriving a basis of  $V_{\text{err}}$ . Even worse, the specific algebraic structure of the deployed code can allow more refined attacks (see our analysis in [1] as an example).

Interestingly, even in this weaker scenario, comparatively efficient schemes can be possible as we will demonstrate in the next section. Furthermore, it seems to be hard problem to figure out the used codes from the observed erroneous codewords, that is to transform the secret code case into the known code case. This problem is known as the *noisy code recognition problem*: the attacker sees noisy codewords of an unknown code, and tries first to find the code, and also to decode. The noisy code recognition problem has been studied in [48, 49, 14], and the associated decision problem recognized to be NP-complete. The general principle for dealing with such a problem is to gather noisy codewords and try finding codewords in the dual code using linear algebra. Of course, due to errors, these words will rather tend to be wrong. However, the lower the weight of words in the dual, the more probably it is for them to be correct words of the dual code of the code.

Due to the limited allowed number of encryptions in the presented scheme, an attacker would not able to collect many codewords, which in particular prevents the low weight dual codewords attack. Furthermore, we discuss [1] concrete instantiation how parameters can be chosen such that the expected number of low weight codewords in the dual codeword is negligible.

## 5 A Concrete Instantiation based on Reed-Muller Codes

In this part, we present a concrete instantiation of our scheme based on punctured Reed-Muller (RM) codes. RM codes are based on evaluating multivariate polynomials on certain points. Adopting the notation from Def. 2, the geometric object  $\mathcal{X}$  is a vector space  $\mathbb{F}^t$  over a finite field  $\mathbb{F}$  of size  $q$ . The vector space

$\mathcal{L}_{t,\rho}$  of functions is obtained by taking multivariate polynomials in  $\mathbb{F}[v_1, \dots, v_t]$  of total degree strictly less than  $\rho$  with coefficients over  $\mathbb{F}$ :  $\mathcal{L}_{t,\rho} := \{f \in \mathbb{F}[v_1, \dots, v_t] \mid \deg(f) < \rho\}$ . The code support  $\mathbf{x} = (x_1, \dots, x_n)$  is a vector of  $n$  distinct elements of  $\mathbb{F}^t$ . The  $q$ -ary RM code of order  $\rho < q$ , denoted by  $\text{RM}_q(t, \rho)$ , is defined by  $\text{RM}_q(t, \rho) := \{(f(x_1), \dots, f(x_n)) \in \mathbb{F}^n \mid f \in \mathcal{L}_{t,\rho}\}$  where  $q$  denotes the size of the deployed field. The usual *full* RM code is obtained when the code support  $\mathbf{x} = (x_1, \dots, x_{q^t})$  contains all elements of  $\mathbb{F}^t$ . For the sake of efficiency and security, we consider *punctured* RM codes where  $n < q^t$  (i.e. some positions are removed). In the sequel, punctured codes are denoted by  $\text{RM}_q(t, \rho)^*$ .

In what follows, we consider only RM codes that are special evaluation codes, that is where the encoding is realized as specified in Def. 4. Furthermore, we have

**Corollary 1.** *Let  $1 \leq \mu \leq q$  be arbitrary. Then there exist RM codes  $\check{\mathcal{C}}, \mathcal{C}, \hat{\mathcal{C}}$  such that (i)  $\mathcal{C}^\mu \subseteq \hat{\mathcal{C}}$  and (ii)  $\check{\mathcal{C}}^2 \subseteq \mathcal{C}$  and  $\check{\mathcal{C}}$  allows for a special encoding of 0.*

Thus, RM codes can be used to instantiate our scheme (proof is given in [1]).

In [1], we present a possible approach for instantiating RM codes such that the scheme works correctly and the DSCP seems to be hard. Shortly summed up, the length  $n$  is in  $\mathcal{O}(\mu^3 \cdot s)$  and the field size (in bits)  $\log_2 q$  is in  $\mathcal{O}(s^{2/3})$ . This yields a ciphertext length in bits in  $\mathcal{O}(\mu^3 \cdot s^{5/3})$ . Furthermore, it holds that  $\frac{T \cdot L \cdot (n - T + 3)}{|\mathbb{F}|} \leq s^3 / 2^{s^{2/3}} = \text{negl}(s)$ . Thus, the upper bound given in Th. 2 is negligible in  $s$  which in turn guarantees the semantic security according to Th. 3.

Of course, for practical instantiations concrete values are more meaningful. In Table 1, we list the values  $n_{\min}$  and  $\log_2 q_{\min}$  (referring to the smallest possible choices of  $n$  and  $q$ ) for different selections of  $s$  and  $\mu$ . With these parameters, both the identified attacks on DSCP (see [1]) and the approaches for recovering the code become infeasible. Observe that the choice of these values concerned only the DSCP but not the (non-tight) upper bound in eq. (4). If one wants to choose parameters such that  $T \cdot L \cdot (n - T + 3) / |\mathbb{F}| \leq 2^{-s}$ , then bigger fields need to be chosen. This yields an increase of the ciphertext length by a factor between 6 and 10 (depending on  $s$ ). We implemented our scheme in the algebra system Magma V2.11-1 and did let it repeatedly run for several parameters (on a W500 Thinkpad laptop with 4GB RAM). The results can be found in Table 2. There, "setup" includes the precomputations mentioned in Sec. 3.3. As the estimated effort for the precomputation is in  $\mathcal{O}(n^3) = \mathcal{O}(\mu^9 \cdot s^3)$  it is not surprising that it takes by far the most time. However, our tests indicated that the bottleneck is actually the memory consumption. Observe that the major step of



Security Parameter	$s = 80$	$s = 128$	$s = 256$	$s = 80$	$s = 128$	$s = 256$
$\mu$	$\mu = 2$			$\mu = 3$		
$n_{\min}$	4,725	8,411	19,186	14,236	26,280	61,044
$\log_2(q_{\min})$	17	18	23	18	19	24
Ciphertext length	9.81 KByte	18.48 KByte	53.87 KByte	31.34 KByte	60.95 KByte	178.84 KByte
Key size ( $= n_{\min}$ bits)	591 Byte	1.02 KByte	2.34 KByte	1.74 KByte	3.21 KByte	7.45 KByte
$\mu$	$\mu = 5$			$\mu = 10$		
$n_{\min}$	60,176	114,189	269,327	448,017	862,336	2,076,969
$\log_2(q_{\min})$	20	20	25	22	24	27
Ciphertext length	146.91 KByte	278.78 KByte	821.92 KByte	1.17 MByte	2.47 MByte	6.68 MByte
Key size ( $= n_{\min}$ bits)	7.35 KByte	13.94 KByte	32.88 KByte	54.69 KByte	105.27 KByte	253.54 KByte
$\mu$	$\mu = 100$					
$n_{\min}$	419,217,826	817,560,769	2,008,578,063			
$\log_2(q_{\min})$	29	31	33			
Ciphertext length	1.42 GByte	2.95 GByte	7.72 GByte			
Key size ( $= n_{\min}$ bits)	49.97 MByte	97.46 MByte	239.44 MByte			

**Table 1.** Computation of the minimum length  $n_{\min}$  of the ciphertexts and the minimum field size  $\log_2 q_{\min}$  in respect to the security parameter  $s$  and the number of multiplications..

the precomputation is to find the linear mapping

$$\lambda_I : \mathbb{F}^{|I|} \rightarrow \mathbb{F}, (p(x_i))_{i \in I} \mapsto p(y) \quad (5)$$

which can be done by computing the kernel of a given matrix (where the rows are evaluations of linearly independent polynomials  $p_i$  on the good points). As this matrix possesses a special structure, there might be still room for improvement. Independent of this, the precomputation can be parallelized to some extent (compute the kernels of different submatrices in parallel and intersect them at the end). Furthermore, some kind of ”global” precomputation is possible. More

Parameters	Effort Setup	Effort Encryption	Effort Decryption	Effort Addition	Effort Multiplication
$\mu = 2$ $s = 80$	Min: 1m 57.781 s Max: 1m 58.998s Av: 1m 58.33s	Min: 0.031s Max: 0.11s Av: 0.072s	Min: $< 10^{-28}s$ Max: 0.032s Av: 0.001	Min: $< 10^{-28}s$ Max: 0.016s Av: 0.000573s	Min: $< 10^{-28}s$ Max: 0.032s Av: 0.005238s
$\mu = 2$ $s = 128$	Min: 1h 18m 22.089 s Max: 1h 20m 21.024s Av: 1h 19m 12.149s	Min: 0.686s Max: 1.014s Av: 0.817s	Min: $< 10^{-28}s$ Max: 0.016s Av: 0.004s	Min: $< 10^{-28}s$ Max: 0.031s Av: 0.0017s	Min: $< 10^{-28}s$ Max: 0.032s Av: 0.01044s
$\mu = 3$ $s = 80$	Min: 46m 3.089 s Max: 47m 4.024s Av: 46m 40.149s	Min: 0.171s Max: 0.312s Av: 0.234s	Min: $< 10^{-28}s$ Max: 0.016s Av: 0.002s	Min: $< 10^{-28}s$ Max: 0.016s Av: 0.0015s	Min: $< 10^{-28}s$ Max: 0.047s Av: 0.014s

**Table 2.** Run time of the proposed encryption scheme for different parameters. The entries represent the minimum (Min) time, the maximum (Max) time, and the average (Av) time in the experiments.

precisely, given the mappings  $\lambda_{I_j}$  for sufficient many linearly independent vectors  $I_j \in \mathbb{F}^T$ , one can compute any other mapping  $\lambda_I$  from them. In other words, the results from precomputations done for one key can be used to accelerate the precomputations for another key.

On the good side, it turns out that once the precomputation is accomplished, all other operations (e.g., encryption, decryption, etc.) are comparatively efficient (1 second or less for the considered test cases). Given the novelty of this approach and the fact that the implementation is by no means optimized, we consider these results as quite promising. Especially in comparison with other schemes that provide additive and multiplicative homomorphism (e.g., see the overview in Sec. 1), our scheme is highly efficient.<sup>12</sup>

*Discussion and Conclusions.* We gave the first encryption scheme based on coding theory that is homomorphic wrt. two operations: addition and multiplication. For structural reasons, the scheme has to be secret-key and the number of encryptions needs to be limited. Nonetheless, applications exist where such schemes can be useful. This is in particular true as we could describe concrete instantiations that are comparatively highly efficient.

The most important question is probably whether other code-based constructions exist without the identified limitations. One approach could be to look for more involved constructions where linear codes are not directly represented. However, caution must be paid to not lose the advantages with respect to efficiency, e.g., the linear decryption algorithm. Another approach could be to replace the field  $\mathbb{F}$  by a weaker algebraic structure, e.g., a ring. This certainly avoids the straightforward application of methods from linear algebra. However, one has to pay attention that the decoding procedure, e.g., interpolation of polynomials, is still possible.

Another question is the investigation of the secret code case. Keeping the code secret should add a further security margin, but how much and what is the impact on the parameter sizes? But even for the known code scenario, interesting problems remain. Although we picked Reed-Muller codes for our concrete instantiations, other codes might yield even better results. As also codes with a low correction capacity might be used, the code might be chosen from a wide set of possible codes. Another possible extension is to remove the condition of fixed error locations. This may allow a reduction to a hard decoding problem. Of course, this likewise means that the noise can grow but similar holds for all presented bootstrappable schemes as well.

Another research direction is whether the schemes can be made KDM (key dependent message) secure. Informally, KDM security means that a scheme remains secure even if an encryption of the deployed key is given, that is  $E_k(k)$ , and is crucial for Gentry's bootstrapping technique. However, the KDM security of the proposed bootstrappable schemes is an open question and needs fur-

---

<sup>12</sup> Of course, other schemes may provide a higher flexibility, e.g., being public-key. We focus here only on the case that a symmetric scheme with bounded multiplicative homomorphism and a bounded number of encryptions are sufficient, e.g., the applications discussed in Sec. 3.4.

ther investigation. Interestingly, the proposed scheme (in its "linear-algebra"-description from Sec. 3.2) has some striking similarities with a scheme by Boneh et al. [10]. The latter has been one of the first schemes for which KDM security could be proven using standard assumptions. Indeed, similar arguments may be used here but the underlying problem, the rank problem, is easy in  $\mathbb{F}^n$ . This immediately raises the question whether variants of the scheme may exist for which KDM security can be shown.

## References

1. Frederik Armknecht, Daniel Augot, Ludovic Perret, and Ahmad-Reza Sadeghi. On constructing homomorphic encryption schemes from coding theory. Eprint Report 2011/309, 2011.
2. Frederik Armknecht, Andreas Peter, and Stefan Katzenbeisser. A cleaner view on IND-CCA1 secure homomorphic encryption using SOAP. Cryptology ePrint Archive, Report 2010/501, 2010. <http://eprint.iacr.org/>.
3. R. Avanzi. Lightweight asymmetric cryptography and alternatives to RSA, ecrypt european network of excellence in cryptology ist-2002-507932. <http://www.ecrypt.eu.org/ecrypt1/documents/D.AZTEC.2-1.2.pdf>, 2005.
4. M. Baldi, M. Bodrato, and G.F. Chiaraluca. A new analysis of the McEliece cryptosystem based on QC-LDPC codes. In *Security and Cryptography for Networks (SCN)*, pages 246–262, 2008.
5. J. Benaloh. *Verifiable secret-ballot elections*. PhD thesis, Yale University, New Haven, CT, USA, 1987.
6. D. Bleichenbacher, A. Kiayias, and M. Yung. Decoding of interleaved reed solomon codes over noisy data. In Jos C. M. Baeten, J. Karel Lenstra, Joachim Parrow, and Gerhard J. Woeginger, editors, *ICALP*, volume 2719 of *Lecture Notes in Computer Science*, pages 97–108. Springer, 2003.
7. D. Boneh and R. Lipton. Algorithms for black-box fields and their application to cryptography (extended abstract). In *CRYPTO '96: Proceedings of the 16th Annual International Cryptology Conference on Advances in Cryptology*, pages 283–297, London, UK, 1996. Springer-Verlag.
8. Dan Boneh and David Mandell Freeman. Linearly homomorphic signatures over binary fields and new tools for lattice-based signatures. Cryptology ePrint Archive, Report 2010/453, 2010. Accepted to PKC'11.
9. Dan Boneh and David Mandell Freeman. Homomorphic signatures for polynomial functions. Cryptology ePrint Archive, Report 2011/018, 2011. Accepted to EUROCRYPT'11.
10. Dan Boneh, Shai Halevi, Michael Hamburg, and Rafail Ostrovsky. Circular-secure encryption from decision diffie-hellman. In *CRYPTO*, pages 108–125, 2008.
11. A. Brown, L. Minder, and A. Shokrollahi. Improved decoding of interleaved ag codes. In Nigel P. Smart, editor, *IMA Int. Conf.*, volume 3796 of *Lecture Notes in Computer Science*, pages 37–46. Springer, 2005.
12. Hao Chen and Ronald Cramer. Algebraic geometric secret sharing schemes and secure multiparty computations over small fields. In Cynthia Dwork, editor, *CRYPTO*, volume 4117 of *Lecture Notes in Computer Science*, pages 521–536. Springer, 2006.
13. Hao Chen, Ronald Cramer, Robbert de Haan, and Ignacio Cascudo Pueyo. Strongly multiplicative ramp schemes from high degree rational points on curves. In Nigel P. Smart, editor, *EUROCRYPT*, volume 4965 of *Lecture Notes in Computer Science*, pages 451–470. Springer, 2008.

14. Mathieu Cluzeau, Matthieu Finiasz, and Jean-Pierre Tillich. Methods for the reconstruction of parallel turbo codes. *CoRR*, abs/1006.0259, 2010.
15. J. Cohen and M. Fischer. A robust and verifiable cryptographically secure election scheme (extended abstract). In *FOCS*, pages 372–382. IEEE, 1985.
16. D. Coppersmith and M. Sudan. Reconstructing curves in three (and higher) dimensional space from noisy data. In *STOC '03: Proceedings of the thirty-fifth annual ACM symposium on Theory of computing*, pages 136–142, New York, NY, USA, 2003. ACM.
17. N. T. Courtois, M. Finiasz, and N. Sendrier. How to achieve a McEliece-based digital signature scheme. *Lecture Notes in Computer Science*, 2248:157–174, 2001.
18. R. Cramer, I. Damgaard, and J. Nielsen. Multiparty computation from threshold homomorphic encryption. In *EUROCRYPT '01: Proceedings of the International Conference on the Theory and Application of Cryptographic Techniques*, pages 280–299, London, UK, 2001. Springer-Verlag.
19. R. Cramer, M. Franklin, L. Schoenmakers, and M. Yung. Multi-authority secret-ballot elections with linear work. Technical report, CWI (Centre for Mathematics and Computer Science), Amsterdam, The Netherlands, The Netherlands, 1995.
20. R. Cramer, R. Gennaro, and B. Schoenmakers. A secure and optimally efficient multi-authority election scheme. *European Transactions on Telecommunications*, 8(5):481–490, September 1997.
21. Ronald Cramer. The arithmetic codex: Theory and applications. In Kenneth G. Paterson, editor, *EUROCRYPT*, Lecture Notes in Computer Science, page 1. Springer, 2011. Accepted to CRYPT'11.
22. I. Damgaard and M. Jurik. A generalisation, a simplification and some applications of pailier's probabilistic public-key system. In *PKC '01: Proceedings of the 4th International Workshop on Practice and Theory in Public Key Cryptography*, pages 119–136, London, UK, 2001. Springer-Verlag.
23. M. Fellows and N. Kobitz. Combinatorial cryptosystems galore! *Contemporary Mathematics*, 168:51–61, 1993.
24. T. El Gamal. A public key cryptosystem and a signature scheme based on discrete logarithms. *IEEE Transactions on Information Theory*, 31(4):469–472, 1985.
25. Craig Gentry. *A fully homomorphic encryption scheme*. PhD thesis, Stanford University, 2009. [crypto.stanford.edu/craig](http://crypto.stanford.edu/craig).
26. Craig Gentry. Fully homomorphic encryption using ideal lattices. In Michael Mitzenmacher, editor, *STOC*, pages 169–178. ACM, 2009.
27. Craig Gentry and Shai Halevi. Implementing Gentry's fully-homomorphic encryption scheme. Accepted to EUROCRYPT'11, 2011.
28. Craig Gentry, Shai Halevi, and Vinod Vaikuntanathan. *i*-hop homomorphic encryption and rerandomizable yao circuits. In Rabin [45], pages 155–172.
29. S. Goldwasser and S. Micali. Probabilistic encryption. *Journal of Computer and System Sciences*, 28:270–299, 1984.
30. Shai Halevi, editor. *Advances in Cryptology - CRYPTO 2009, 29th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 16-20, 2009. Proceedings*, volume 5677 of *Lecture Notes in Computer Science*. Springer, 2009.
31. T. Høholdt, J. H. van Lint, and R. Pellikaan. *Handbook of Coding Theory*, volume I, chapter Algebraic geometry codes, pages 871–961. Elsevier, 1998.
32. A. Kiayias and M. Yung. Cryptographic hardness based on the decoding of Reed-Solomon codes with applications. *Electronic Colloquium on Computational Complexity (ECCC)*, 2002.
33. A. Kiayias and M. Yung. Cryptographic hardness based on the decoding of Reed-Solomon codes. Cryptology ePrint Archive, Report 2007/153, 2007. <http://eprint.iacr.org/>.

34. A. Kiayias and M. Yung. Cryptographic hardness based on the decoding of reed-solomon codes. *IEEE Transactions on Information Theory*, 54(6):2752–2769, 2008.
35. E. Kushilevitz and R. Ostrovsky. Replication is not needed: single database, computationally-private information retrieval. In *FOCS '97: Proceedings of the 38th Annual Symposium on Foundations of Computer Science (FOCS '97)*, page 364, Washington, DC, USA, 1997. IEEE Computer Society.
36. F. J. Macwilliams and N. J. A. Sloane. *The Theory of Error-Correcting Codes*. North-Holland Mathematical Library. North Holland, January 1983.
37. R. J. McEliece. *A Public-Key System Based on Algebraic Coding Theory*, pages 114–116. Jet Propulsion Lab, 1978. DSN Progress Report 44.
38. Carlos Aguilar Melchor, Philippe Gaborit, and Javier Herranz. Additively homomorphic encryption with  $\ell$ -operand multiplications. In Rabin [45], pages 138–154.
39. M. Naor and B. Pinkas. Oblivious polynomial evaluation. *SIAM J. Comput.*, 35(5):1254–1281, 2006.
40. H. Niederreiter. A public-key cryptosystem based on shift register sequences. In *EUROCRYPT*, volume 219 of *LNCS*, pages 35–39, 1985.
41. P. Paillier. Public-key cryptosystems based on composite degree residuosity classes. In *EUROCRYPT*, pages 223–238, 1999.
42. Manoj Prabhakaran and Mike Rosulek. Homomorphic encryption with cca security. In *ICALP '08: Proceedings of the 35th international colloquium on Automata, Languages and Programming, Part II*, pages 667–678, Berlin, Heidelberg, 2008. Springer-Verlag.
43. Ignacio Cascudo Pueyo, Hao Chen, Ronald Cramer, and Chaoping Xing. Asymptotically good ideal linear secret sharing with strong multiplication over  $n$ -fixed finite field. In Halevi [30], pages 466–486.
44. Ignacio Cascudo Pueyo, Ronald Cramer, and Chaoping Xing. The torsion-limit for algebraic function fields and its application to arithmetic secret sharing. In Halevi [30], pages 466–486.
45. Tal Rabin, editor. *Advances in Cryptology - CRYPTO 2010, 30th Annual Cryptology Conference, Santa Barbara, CA, USA, August 15-19, 2010. Proceedings*, volume 6223 of *Lecture Notes in Computer Science*. Springer, 2010.
46. Nigel P. Smart and Frederik Vercauteren. Fully homomorphic encryption with relatively small key and ciphertext sizes. In Phong Q. Nguyen and David Pointcheval, editors, *Public Key Cryptography*, volume 6056 of *Lecture Notes in Computer Science*, pages 420–443. Springer, 2010.
47. Damien Stehlé and Ron Steinfeld. Faster fully homomorphic encryption. In Masayuki Abe, editor, *ASIACRYPT*, volume 6477 of *Lecture Notes in Computer Science*, pages 377–394. Springer, 2010.
48. Antoine Valembois. Detection and recognition of a binary linear code. *Discrete Applied Mathematics*, 111(1-2):199–218, 2001.
49. Antoine Valembois. *Décodage, Détection et Reconnaissance des Codes Linéaires Binaires*. PhD thesis, Université Limoges, 2004.
50. Marten van Dijk, Craig Gentry, Shai Halevi, and Vinod Vaikuntanathan. Fully homomorphic encryption over the integers. In Henri Gilbert, editor, *EUROCRYPT*, volume 6110 of *Lecture Notes in Computer Science*, pages 24–43. Springer, 2010.